1-1-2016

High Altitude Payload for CubeSat Aeroboom Development

Jessica Gardiner
Utah State University

High Altitude Payload for CubeSat Aeroboom Development

HAPCAD

Physics 4900 Research
Jessica Gardiner, Undergraduate Researcher
Jan Sojka, Faculty Mentor
Utah State Department of Physics
January, 2016

Abstract

The intent of this project was to create and launch a high altitude balloon flight for the continued research of the Utah State University CubeSat project. The High Altitude Payload for CubeSat Aeroboom Development (HAPCAD) is a direct support research project for the Get Away Special Passive Attitude Control Satellite (GASPACS). HAPCAD validated the aeroboom design and deployment mechanism by deploying an aeroboom in a near space environment. It was essential for the GASPACS payload to deploy in an environmental pressure lower than the internal aeroboom pressure for experimental success; this pressure is approximately 100 *torr*. The most effective time to deploy the aeroboom was determined to be 75,000 feet. A successful deployment was recorded on August 1, 2015, at approximately 36,000 feet. Due to a slow ascent rate, the aeroboom deployed at a low altitude and did not inflate. However, the mission still obtained an automated successful aeroboom deployment. Follow up experiments will be scheduled once testing on a new aeroboom system is complete.

Introduction

The Get Away Special (GAS) Team at Utah State University is in the process of building a cube satellite, known as a CubeSat, which will test passive attitude control using aerodynamic torque in Low Earth Orbit, LEO (Shrivastava & Modi, 1983). This mission is called the Get Away Special Passive Attitude Control Satellite, or GASPACS. The passive control will be obtained through the use of an extended aeroboom from one end of the satellite after launch into low earth orbit (LEO). This aeroboom will be deployed using residual air and a nichrome cut wire assembly, and rigidized by a UV curable epoxy coated along the length of the aeroboom. The purpose of the High Altitude Payload for CubeSat Aeroboom Development, or HAPCAD, was to verify the aeroboom design and the deployment mechanism for GASPACS in a near space environment. The near space environment provided a higher level of testing, in which the conditions were more space-like than earth. The near-space environment provided low pressure, variable temperatures, better UV light, and an opportunity for the system to be fully independent. Successful tests increase the technology readiness level of GASPACS. The High Altitude Reconnaissance Balloon for Outreach and Research (HARBOR) from Weber State provides undergraduate research opportunities to students for projects in a near-space environment (Sohl, 2016). Most projects involve atmospheric research, however the near-space environment provided adequate conditions for GASPACS verification before higher level analysis. By collaborating with the HARBOR program of Weber State University the GAS team gained an opportunity to verify the design of GASPACS by testing HAPCAD.

Theory

Residual air inflation can be described using simple physics. On earth, standard pressure is 14.7 *psi*, or approximately 760 *Torr* (Schroeder, 2000). In LEO, the pressure is essentially zero, in terms of comparing the pressure on earth and the pressure in LEO. The aeroboom inflates from the force of the pressure gradient between the small amount of air contained within the aeroboom and the vacuum of space. The pressure gradient force can be described by the following equation (Gardiner, 2016), (Panoff, 2016);

$$-\triangle f = -\frac{1}{\rho}\frac{\Delta P}{\Delta x} \tag{1}$$

Where $\triangle f$ is the pressure gradient per unit mass, ρ is the gas density, P is the pressure, and x is the thickness of the aeroboom wall, the pressure gradient is the inverse of the gas density multiplied by the difference in pressure over the thickness of the aeroboom wall (Gardiner, 2016). It can be concluded that a minimal initial pressure inside the aeroboom will expand in LEO due to the changing pressure gradient. A minimal pressure inside the aeroboom is ideal, in order to minimize the area occupied by the compressed boom, this pressure was experimentally determined to be 2.2 *psi* of gas[1] at STP (Gardiner, 2016), and yields a compressed, flat aeroboom on earth.

The physical mechanism used to deploy the aeroboom was a nickel chromium heating wire assembly (Gardiner, 2016). In Figure (1), the left picture shows the nichrome wire assembly. Threaded through the entire container and nichrome loop is Dyneema[2] fishing line, and can be seen in the right picture in Figure (1). An additional nichrome wire assembly is also attached to the container as an alternate deployment mechanism, in the event of the first nichrome loop failing. Upon melting the thread, the aeroboom is released from the container, being pushed out by the expanding volume of the stored gas. The thread is attached to the container in a way as to not create any debris. Once deployed, the aeroboom hardens in UV light through a UV curable epoxy coated along the aeroboom. The curing process takes approximately 30 seconds.

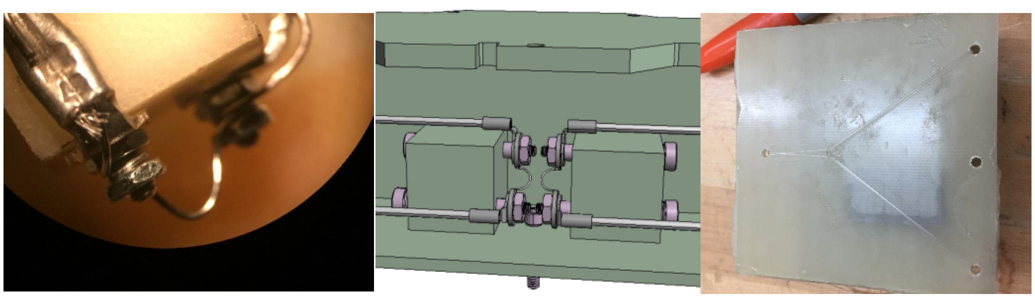

Figure (1) – Left: Nichrome wire assembly. Middle: Diagram of nichrome wire assembly. Right: Dyneema thread assembly.

[1] The gas for the GASPACS mission has yet to be determined. Ambient air was used in the HAPCAD aeroboom.
[2] Dyneema ® is a registered product of DSM – Bright Science. Brighter Living ™

Equations

The following equations were used to determine the expected pressure at a given altitude in the high altitude environment. (Hall, 2015)

For altitudes lower than 36,152 feet:

$$T = 59 - 0.00356h \quad \& \quad p = 2116\left[\frac{T+459.7}{518.6}\right]^{5.256} \tag{2}$$

For altitudes between 36,152 feet and 82,345 feet:

$$T = -70 \quad \& \quad p = 473.1e^{(1.73-0.000048h)} \tag{3}$$

For altitudes greater than 82,345 feet:

$$T = -205.05 + 0.00164h \quad \& \quad p = 51.97\left[\frac{T+459.7}{389.98}\right]^{-11.388} \tag{4}$$

Where *T* is the local temperature in Fahrenheit, *h* is the altitude in feet, and *p* is pressure in psf. Using pressures calculated from altitudes[3] and aeroboom lab testing results, the optimal altitude was determined for successful aeroboom inflation.

Procedure

Ten past HARBOR flights data sets were obtained and examined for pressure, altitude and time measurements. Total flight time was recorded and a graph of altitude vs. time was created for all flights. Additionally, a histogram was created to determine the most probable burst time and altitude. With respect to aeroboom testing, team members built an aeroboom and allowed it to expand in a vacuum chamber several times, taking an average pressure where the aeroboom was considered “fully inflated”. This pressure was used to calculate the initial volume inside the aeroboom and used to create the flight profile.

Flight Profile

The pre-flight research yielded the following results; the maximum external pressure in which the aeroboom inflated inside the vacuum chamber was 100 *Torr.* The majority of high altitude balloon flights are 5500-6000 seconds in length, which is approximately an hour and a half, see Figure (2). The majority of high altitude balloon flights burst after 90,000 *feet*, of see Figure (3), and an average burst altitude of approximately 87,000 *feet*.

[3] See Appendix B for pressures based on altitude

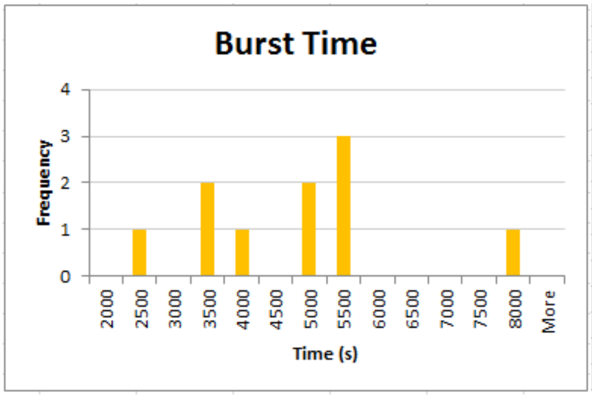

Figure (2) – Histogram depicting the burst time of HARBOR flights.

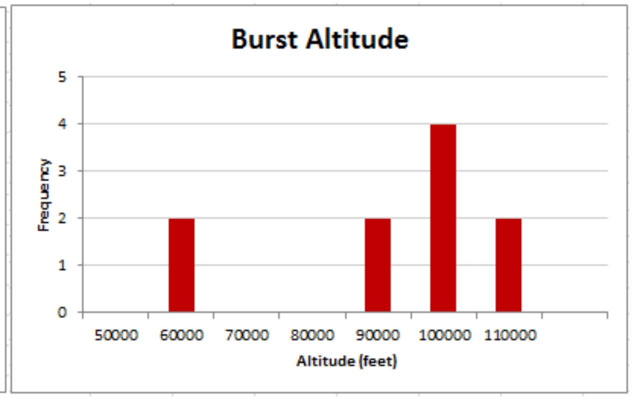

Figure (3) – Histogram depicting the burst altitude of HARBOR flights.

Figure (4) depicts the flights ascension rate for each HARBOR flight studied. The slope of each flight was recorded and an average value was calculated to be 18.29 feet per second with a variance of 16.24 feet per second and a standard deviation of 4.03 feet per second. The variance is quite high, and may be contributed to the small sample size of ten. However it can still be concluded that the majority of flights climb at a rate between 14.26 and 22.32 feet per second. Additionally, each linear model for individual flights represents 99% of the data, and illustrates the consistency in the speed of the balloon for this range of altitudes[4].

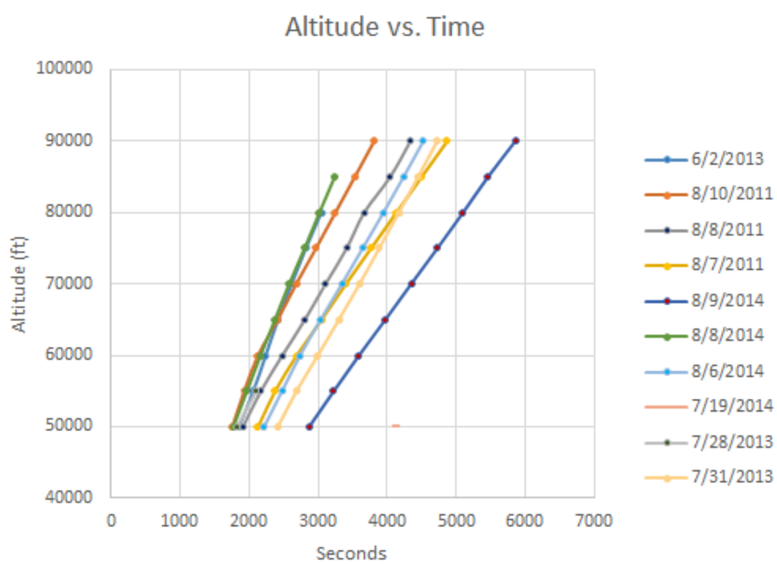

Figure (4) – Lined scatter plot depicting the climb rate of HARBOR flights.

Finally, Figure (5) was created using equations (2), (3), and (4). The graph indicates the aeroboom could be deployed as early as 47,000 feet, at 100 *Torr*.

[4] See Appendix A for HARBOR data tables

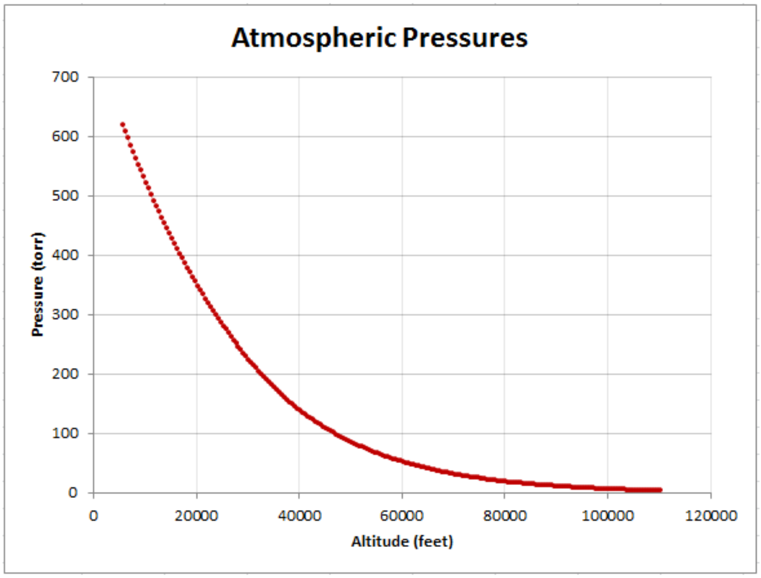

Figure (5) – Scatter plot depicting pressure at given altitudes.

To facilitate a more space-like environment, a lower pressure, the ideal HAPCAD aeroboom deployment was set at 75,000 feet, which has an atmospheric pressure of approximately 26 *Torr*.

HAPCAD Design

The HAPCAD was built inside a plastic project box with dimensions of 17.5 x 12.5 x 7.5 *cm*, the aeroboom was attached to the outside of the project box, and the box was surrounded by ½ *inch* Styrofoam for insulation. Version #1 of HAPCAD used HARBOR's multi-sensor array (MSA) to trigger the nichrome wire assembly. The MSA was programmed[5] by a HARBOR team member to send a signal to HAPCAD once the designated altitude was reached, as measured by a GPS. A Nucleo microcontroller on HAPCAD was coded to begin the aeroboom deployment process after a signal from the MSA was received (NUCLEO-F401RE Datasheet, 2015). Version #1 also contained an aeroboom temperature sensor, an SD card breakout board to save data, and an external indicator light programmed to blink when system was live. HAPCAD was mounted to a camera frame with the aeroboom container on the bottom. The aeroboom container was designed to block UV light from entering the container and causing early curing of the aeroboom. Three cameras were attached to each corner of the camera frame and positioned to view the aeroboom and HAPCAD payload, one still and two GoPros®. The still camera regularly took a picture[6] and the GoPros® took video for the duration of the flight and landing. Figure (6) depicts the expected HARBOR flight manifest. In reality there were several additional payloads between the HARBOR MSA and the parachute. It was essential for HAPCAD to be positioned at the bottom in order for a successful aeroboom deployment. Figure (7) shows the actual payload design.

[5] See Appendix D for Nucleo Code
[6] The exact rate the camera took pictures will be added when the information is received

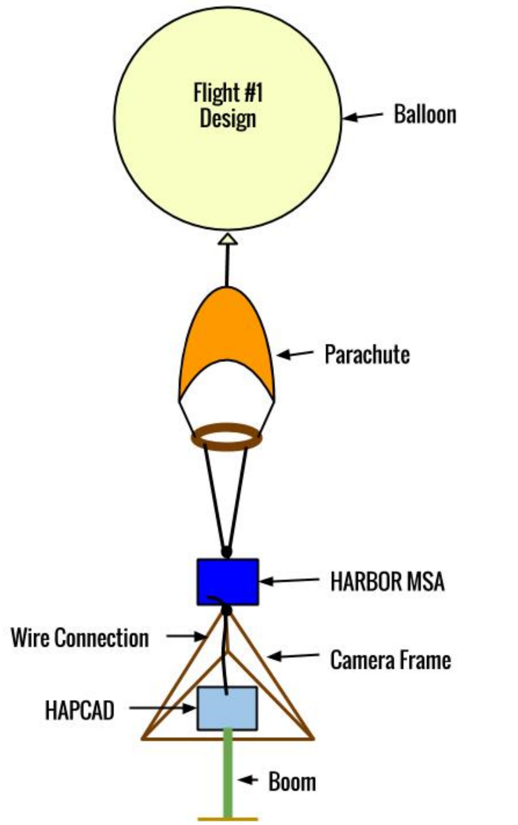

Figure (6) – HAPCAD Version #1 flight manifest.

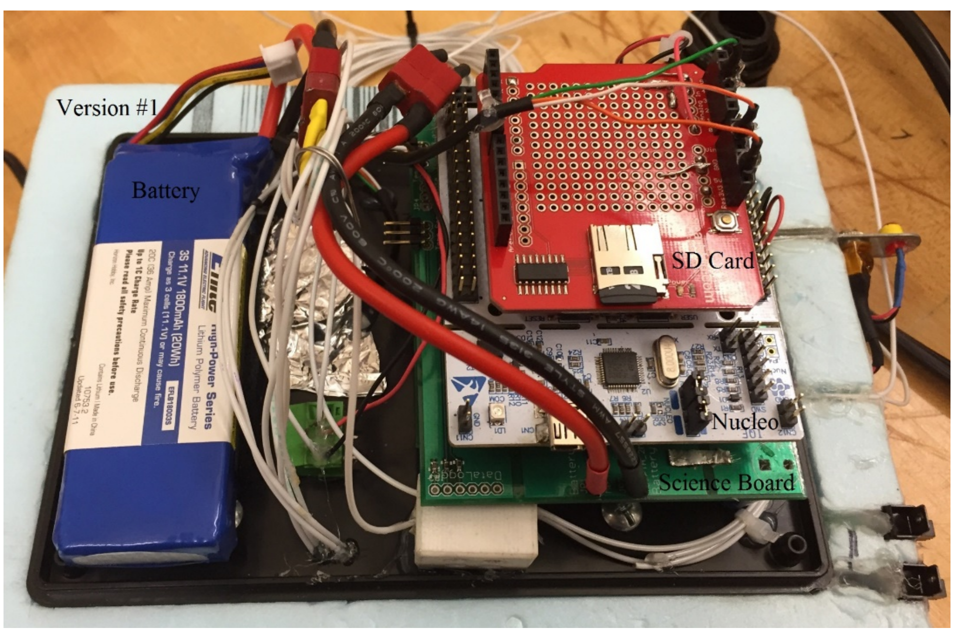

Figure (7) – HAPCAD Version #1 flight design.

Version #2 of HAPCAD was an independent system which used a GPS to trigger the nichrome wire at 75,000 feet, with a real-time clock (RTC) from the Nucleo as a back-up trigger, set to one hour (NUCLEO-F401RE Datasheet, 2015). Additionally, version #2 contained an external temperature sensor, another aeroboom temperature sensor for a total of two temperature sensors, one on each side of the aeroboom box, an upgraded remove before flight pin which was rotated to be flush with the exterior, an exterior communication serial point, allowing team members to see the working code while containing the system within the Styrofoam box, the nichrome wire connection points were rewired to remove failure points, and the SD card was upgraded to an OpenLog™ SD card data saver, which was able to save the data file after every data point was collected[7], see figures (8) and (9).

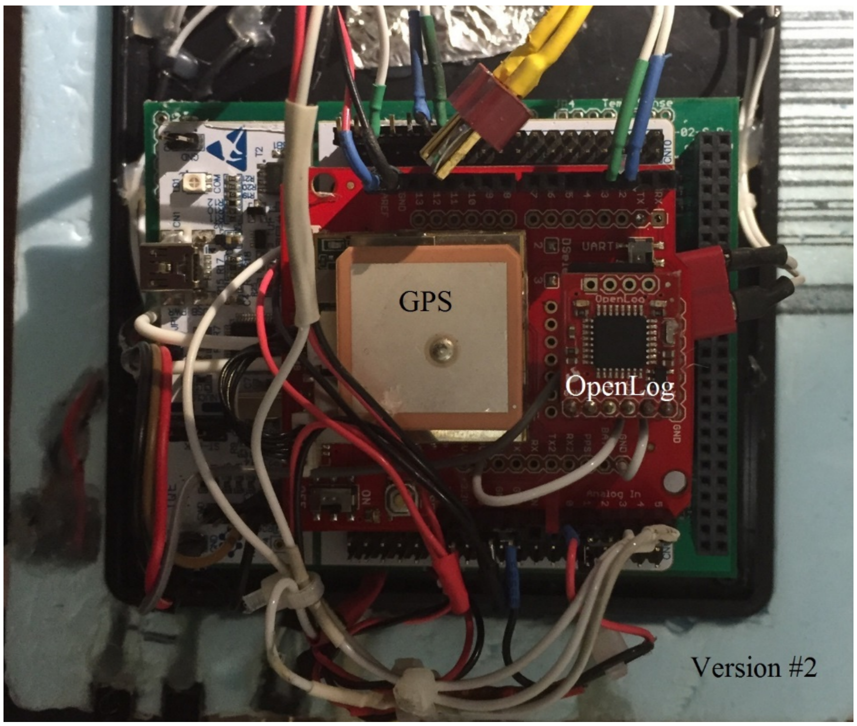

Figure (8) – HAPCAD version #2 internal components.

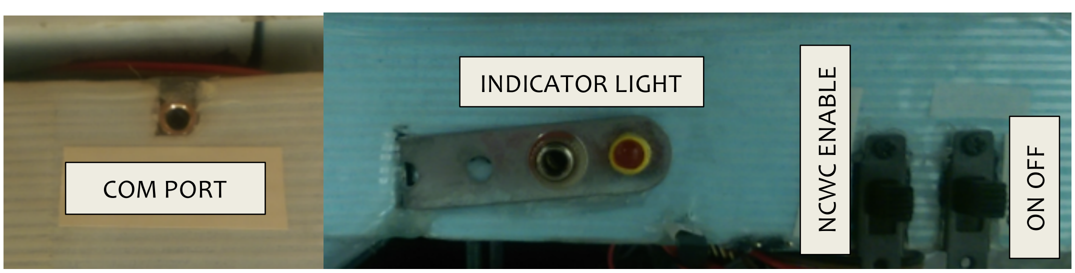

Figure (9) – HAPCAD version #2 external components.

[7] See Appendix C for full list of HAPCAD components

Flight Results

Version #1 flew on June 13th, 2015. HAPCAD and HARBOR MSA reset at beginning of flight. No data was recovered and aeroboom did not deploy. However the high altitude balloon was able to reach a peak altitude before burst[8], See figure (10).

Figure (10) – HARBOR and HAPCAD flight on 6/13/15 Left: immediately after launch. Right: 82,000 feet, immediately before burst.

Version #2 flew on August 1st, 2015. The high altitude balloon rose approximately 2/3 slower than expected and caused the backup timer to deploy the aeroboom at 36,000 *feet*, 39,000 *feet* before the optimal altitude and deployment triggered by the GPS. The low altitude deployment caused the aeroboom to not inflate, however the deployment was successful and the balloon was able to reach peak altitude before getting cut down and 82,000 *feet*, see Figure (11).

[8] The peak altitude and burst time of flight #1 will be added when the information is received

Figure (11) –HAPCAD flight #2 on 8/1/15 Left: immediately after deployment at 36,000 feet. Right: immediately before burst.

All data for version #2 was recorded from the initial start moment up to 1 minute and 48 seconds after aeroboom deployment. All hardware was then shut down to conserve battery. In Figure (12), the temperature of the system is represented. The yellow line represents the downward trend of the external temperature sensor. The zig-zag is most likely caused by the payload spinning and position in relation to the sun. The aeroboom and internal sensors were insulated by Styrofoam, and the sudden temperature drop can be explained by the aeroboom deployment and loss of insulation from the end plate.

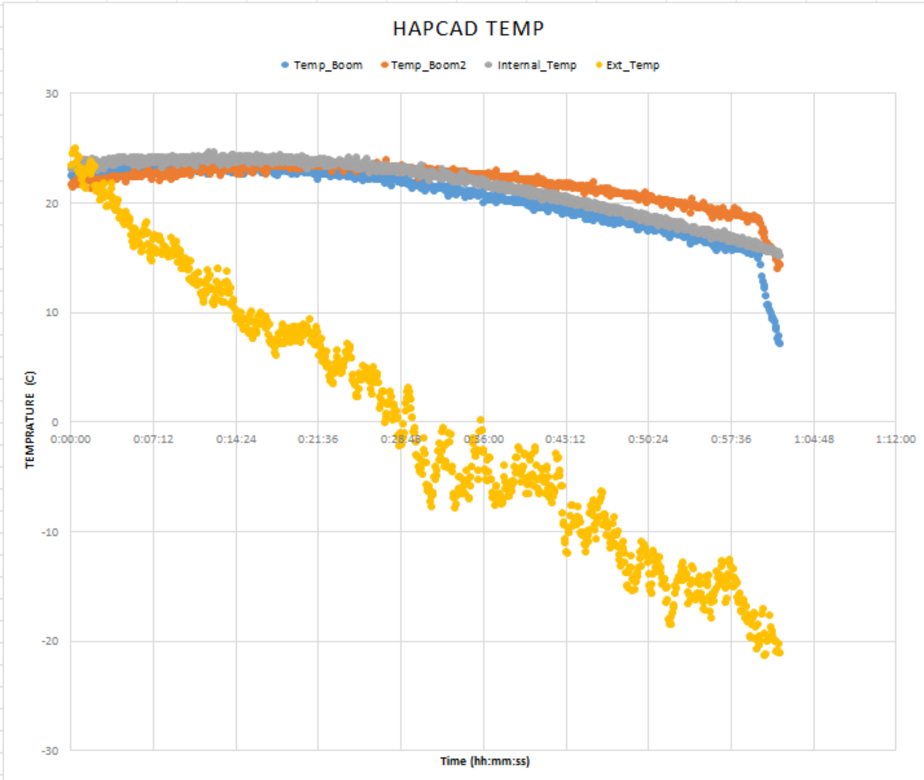

Figure (12) – HAPCAD version #2 temperature profile.

Figure (13) depicts the number of satellites during the flight. The amount remained roughly constant at nine satellites. A minimum of three satellites were required for a GPS lock.

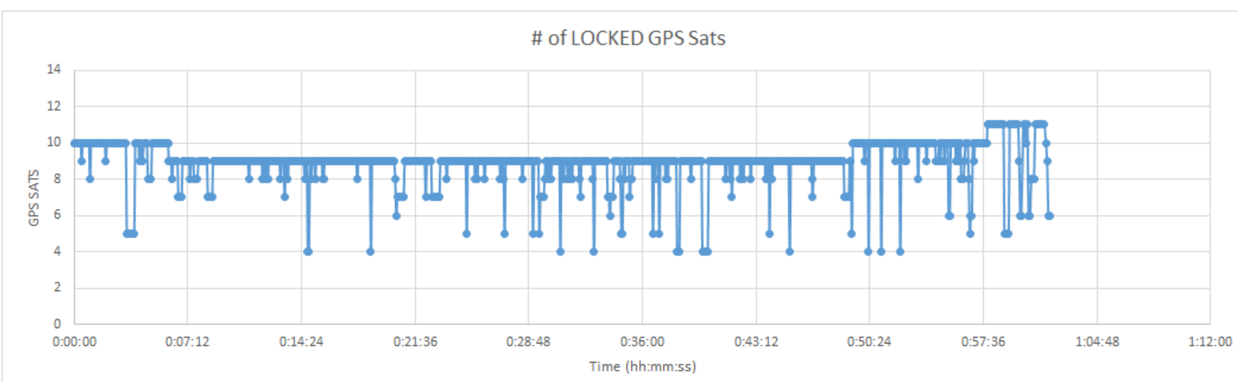

Figure (13) – HAPCAD version #2 GPS profile.

The climb rate of flight #2 is represented in Figure (14). The altitude is based on the GPS and the time on the RTC. A linear model was fitted with the equation $y = 8.3338x + 5626.6$. The slope indicates the average climbing rate in feet per second.

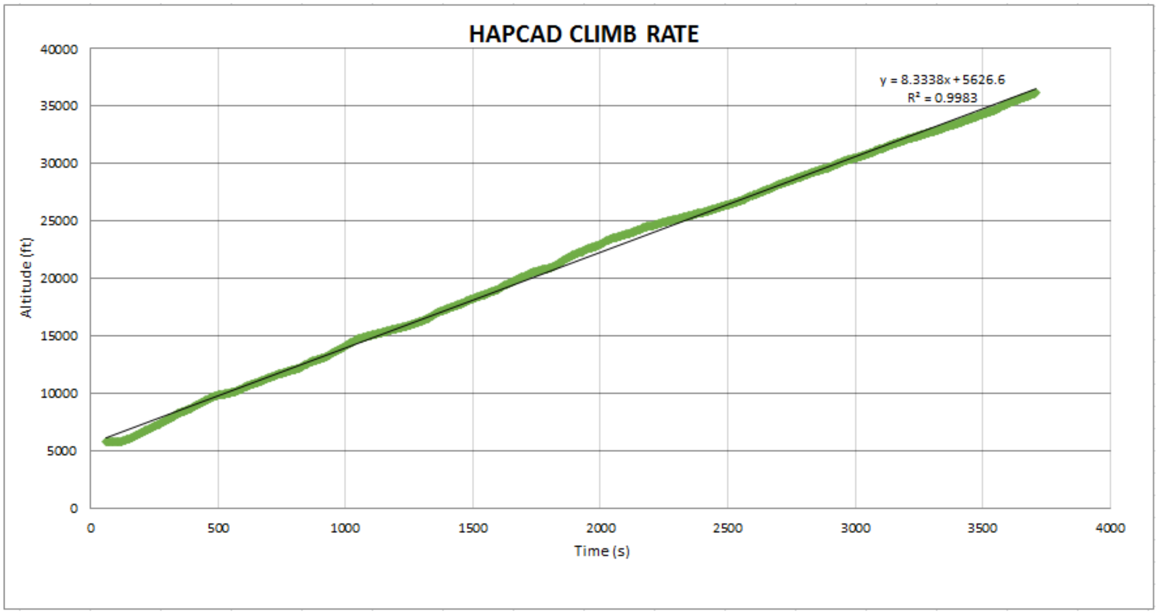

Figure (14) – HAPCAD version #2 rate of ascension.

Figure (15) represents the real time clock and potential drift due to temperature changes. The linear slope indicates the real time clock maintained correct functioning for the duration of the flight. There is a small slope change right after count 900. This is due to the aeroboom deployment and sharp drop in temperature. However, this drift does not affect the GPS calculations enough to warrant a correction.

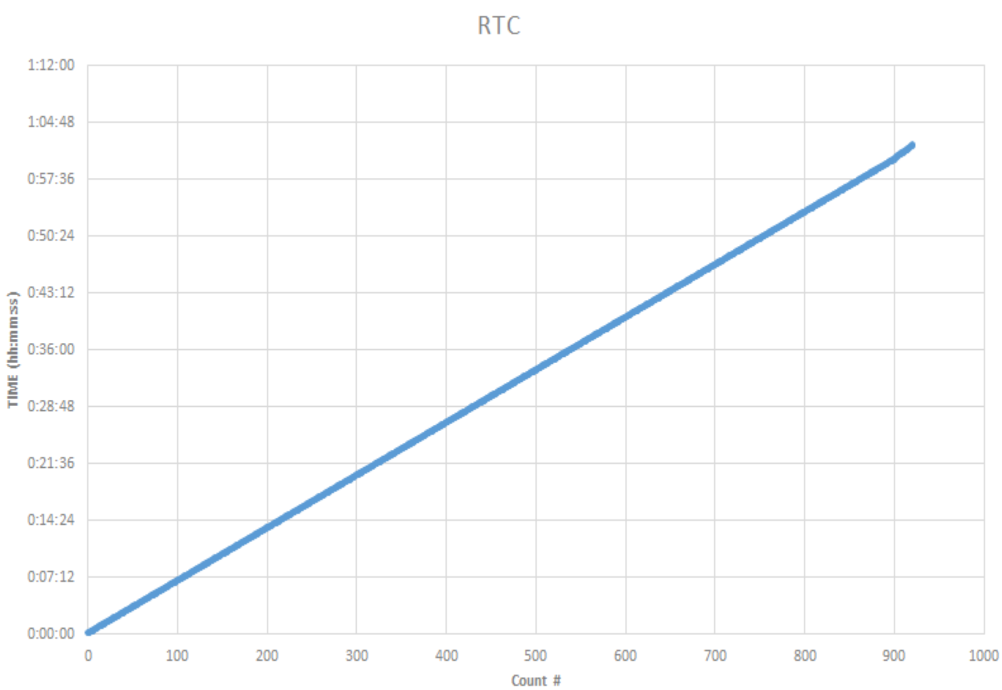

Figure (15) – HAPCAD version #2 RTC profile.

Figure (16) depicts the cured aeroboom taken at HAPCAD payload recovery, approximately four and a half hours after launch.

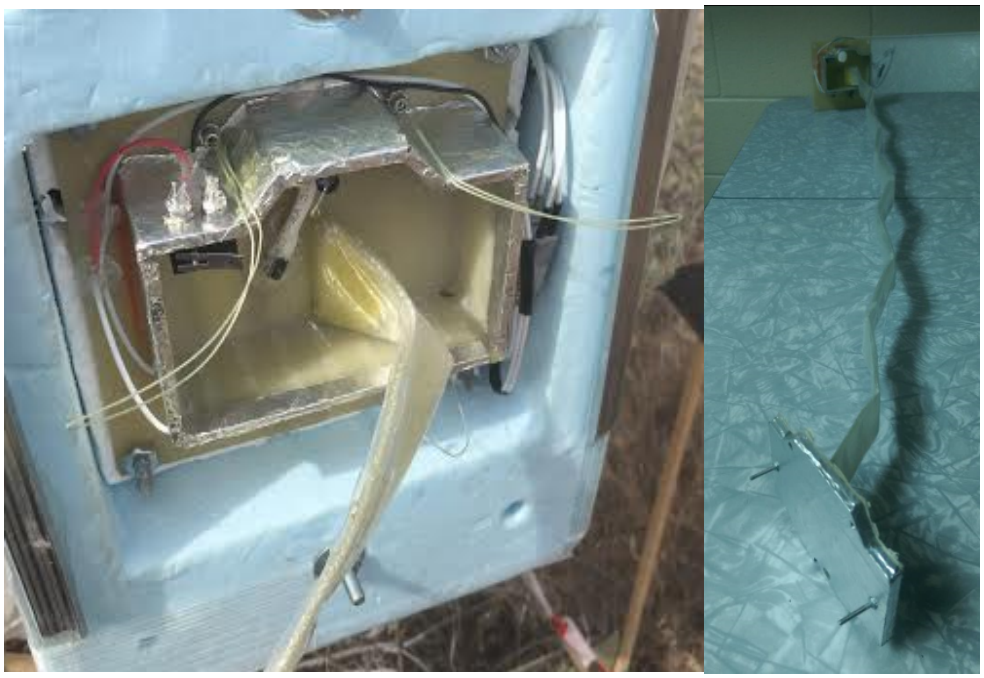

Figure (16) – HAPCAD version #2 deployed and cured aeroboom. Right: at payload recovery. Left: Post-flight inspection

Levels of success were determined by the visual inspection of the aeroboom and inflation via the GoPro cameras on the flight frame. The aeroboom did not appear to possess any major cracks along the resin, nor any holes along the aeroboom. Additionally, the altitude in which the aeroboom deployed was used to determine the pressure inside the aeroboom at deployment, using the values from Figure (5). At 36,000 *feet*, atmospheric pressure is approximately 171 *Torr*[9]. According to the pre-flight research, the highest pressure in which the aeroboom could inflate was 100 *Torr*.

Conclusion

As a summary, HAPCAD Flight #1 verified the aeroboom container design, evidenced by a non-hardened aeroboom at the end of the flight. On HAPCAD Flight #2 the aeroboom deployed at 36,000 *feet* due to an ascent rate 54% slower than the predicted value[10]. The aeroboom did not inflate at due to the high external pressure of 171 *Torr*. Flight #2 validated an automated aeroboom deployment procedure, aeroboom thermal protection options, the nichrome wire system in variable temperatures, and the overall aeroboom design. The deployed aeroboom was built in March, 2015, and flew on flight #1. In the way of hardware, the RTC did not experience

[9] The horizontal position accuracy of the GPS was less than 2.5m

[10] The percent error was calculated as follows: $\frac{|18.29-8.3338|}{18.29} * 100 = 54.41\%$

significant drift and the GPS was sufficiently accurate for the purposes of HAPCAD. And finally, the aeroboom did not tangle during deployment, did not experience additional inflation as the altitude rose, nor did it collapse at balloon burst, all validating the aeroboom design. However, one hardware reservation to consider is that although the aeroboom successfully inflated during each pre-flight test, the level of inflation had decreased over time, indicating that the aeroboom experienced a small amount of leaking due to the repeated stress of inflation, but not enough to cause the aeroboom to not inflate during the flight. The effectiveness of the HAPCAD flight for the purpose of testing the GASPACS experiment must also consider the following limitations; high altitude flights are unable to model gravity, the temperature at high altitude is not an effective model of LEO temperatures, where the range of high altitude temperature is -70°F to 23.4°F [11] (Hall, 2015), and the range of LEO temperatures is -58°F to 176°F (Gardiner, 2016), and also, the ambient pressure in LEO is between $10^{-10}$and $5 * 10^{-8}$ *torr*, which is several orders of magnitude lower than that of the pressures at high altitudes (Christopher J. Scolese , 2007). It can be concluded that using high altitudes to test the HAPCAD system affects the significance of the aeroboom results, but only minimally. The GASPACS aeroboom is a valid scientific concept and design. Further, controlled, low pressure and temperature testing is required using the GASPACS system to further validate the aeroboom deployment procedure. Possible experiments include temperature controlled vacuum chamber testing to further explore aeroboom inflation and deployment properties. Additionally, high altitude rockets offer viable GASPACS tests, as the rocket, in addition to experiencing more accurate LEO conditions, experiences a brief moment of microgravity, and provides an opportunity to deploy the aeroboom in its intended conditions. Use of high altitude balloon flights for prospective small satellites has the potential to be extremely helpful, but only for cases where satellite equipment, such as cameras, is tested. The high altitude environment cannot be used for temperature and pressure sensitive testing as the differences in the environment properties yield inconclusive results. Future goals of the GASPACS experiment include addressing the leaking and sealing issues in the aeroboom, and controlled testing in near space environments to validate the aeroboom deployment procedure.

[11] High altitude temperature ranges are from 10,000 feet to 110,000 feet, see appendix B.

References

Christopher J. Scolese . (2007, 6 3). *Low Earth Orbit Spacecraft Charging Design Handbook.* Retrieved 3 18, 2016, from NASA Technical Standards System: file:///C:/Users/Jessica/Downloads/NASA-HDBK-4006.pdf

Gardiner, J. (2016). *Self-Inflating Deployable Stuctures for use as a CubeSat Passive Attitude Control System.* Utah State University, Physics. Logan: Utah State University. Retrieved 2016

Hall, N. (Ed.). (2015). *Earth Atmosphere Model*. Retrieved 2016, from National Aeronautics and Space Administration: https://www.grc.nasa.gov/www/K-12/airplane/atmos.html

Panoff, R. M. (Ed.). (2016). *Pressure Gradient Force Calculator*. (National Science Digital Library) Retrieved 2016, from SHODOR: A National Resource for Computational Science Education: http://www.shodor.org/os411/courses/_master/tools/calculators/pgf/

Schroeder, D. V. (2000). *Thermal Physics.* Addison Wesley Longman.

Shrivastava, S., & Modi, V. (1983). Satellite attitude dynamics and control in the presence of environmental torques - A brief survey. *Journal of Guidance, Control, and Dynamics, 6*(6), 461-471. doi:10.2514/3.8526

Sohl, J. (2016). (Weber State University) Retrieved 2016, from HARBOR, High Altitude Reconnaissance Balloon for Outreach and Research: http://planet.weber.edu/harbor/

STM32F401xD STM32F401xE. (2015). (3). STMicroelectronics. Retrieved 2016, from ARMmbed: http://www.st.com/st-web-ui/static/active/en/resource/technical/document/datasheet/DM00102166.pdf

Appendix A: HARBOR Flight Data

HARBOR DATA			
Flight	Time (s)	Altitude (feet)	Pressure (torr)
6/2/13	1854	50000	87.5
	2062	55000	68.9
	2242	60000	54.3
	2418	65000	42.7
	2616	70000	33.7
	2823	75000	26.4
	3045	80000	21
		85000	16.1
		90000	13.2
8/10/11	1758	50000	87.5
	1931	55000	68.9
	2130	60000	54.3
	2414	65000	42.7
	2682	70000	33.7
	2970	75000	26.4
	3249	80000	21
	3530	85000	16.1
	3811	90000	13.2
8/8/11	1914	50000	87.5
	2171	55000	68.9
	2483	60000	54.3
	2807	65000	42.7
	3107	70000	33.7
	3414	75000	26.4
	3666	80000	21
	4043	85000	16.1
	4335	90000	13.2
8/7/11	2132	50000	87.5
	2378	55000	68.9
	2691	60000	54.3
	3050	65000	42.7
	3404	70000	33.7
	3767	75000	26.4
	4133	80000	21
	4498	85000	16.1
	4869	90000	13.2
8/9/14	2870	50000	87.5
	3226	55000	68.9
	3592	60000	54.3
	3973	65000	42.7
	4348	70000	33.7
	4727	75000	26.4
	5098	80000	21
	5458	85000	16.1
	5854	90000	13.2
8/8/14	1774	50000	87.5
	1970	55000	68.9
	2167	60000	54.3
	2374	65000	42.7
	2580	70000	33.7
	2805	75000	26.4
	3019	80000	21
	3240	85000	16.1
		90000	13.2
8/6/14	2219	50000	87.5
	2478	55000	68.9
	2738	60000	54.3
	3035	65000	42.7
	3343	70000	33.7
	3644	75000	26.4
	3946	80000	21
	4237	85000	16.1
	4523	90000	13.2
7/19/14	4081	50000	87.5
		55000	68.9
		60000	54.3
		65000	42.7
		70000	33.7
		75000	26.4
		80000	21
		85000	16.1
		90000	13.2
7/28/13	1826	50000	87.5
	2106	55000	68.9
		60000	54.3
		65000	42.7
		70000	33.7
		75000	26.4
		80000	21
		85000	16.1
		90000	13.2
7/31/13	2416	50000	87.5
	2686	55000	68.9
	2995	60000	54.3
	3300	65000	42.7
	3601	70000	33.7
	3886	75000	26.4
	4166	80000	21
	4448	85000	16.1
	4722	90000	13.2

Appendix A Cont.: HARBOR Flight Data: Burst and Climb Rate

Flight	Burst Time (s)	Altitude (feet)	Equation	R-Squared
6/2/2013	3170	82839.9	y=25.566x + 2693	0.9987
8/10/2011	3974	92815.9	y=18.904 + 18592	0.9958
8/8/2011	4879	98467.2	y=16.405x + 19073	0.9992
8/7/2011	5265	94695.5	y=14.318x + 20806	0.9979
8/9/2014	7628	106835	y=13.393x + 11746	0.9999
8/8/2014	3396	88767.4	y=23.814x + 8176.7	0.9994
8/6/2014	5052	98659.1	y=17.115x + 12640	0.9994
7/19/2014	4874	52674.9	no slope	0
7/28/2013	2263	58471.1	y=17.857x + 17393	1
7/31/2013	5310	100472	y=17.198x + 8431.7	0.9997

Appendix B: Expected Pressure and Temperature from Equations (1), (2), and (3).

For altitudes lower than 36,152 feet:

$$T = 59 - 0.00356h \quad \& \quad p = 2116\left[\frac{T+459.7}{518.6}\right]^{5.256} \tag{1}$$

For altitudes between 36,152 feet and 82,345 feet:

$$T = -70 \quad \& \quad p = 473.1e^{(1.73-0.000048h)} \tag{2}$$

For altitudes greater than 82,345 feet:

$$T = -205.05 + 0.00164h \quad \& \quad p = 51.97\left[\frac{T+459.7}{389.98}\right]^{-11.388} \tag{3}$$

Expected Pressures				
Altitude-Feet	Temperature °F	Pressure-PSF	Pressure-PSI	Pressure-Torr
5500	39.42	1730.2996	12.01596963	621.4046679
6000	37.64	1698.1115	11.79244078	609.8448956
6500	35.86	1666.4099	11.572291	598.4598716
7000	34.08	1635.1893	11.35548107	587.2475677
7500	32.3	1604.444	11.14197209	576.2059722
8000	30.52	1574.1685	10.93172547	565.3330895
8500	28.74	1544.3572	10.72470294	554.6269402
9000	26.96	1515.0048	10.52086654	544.0855613
9500	25.18	1486.1057	10.32017863	533.7070056
10000	23.4	1457.6547	10.12260184	523.4893421
10500	21.62	1429.6463	9.928099168	513.4306557
11000	19.84	1402.0753	9.73663387	503.5290469
11500	18.06	1374.9364	9.548169529	493.7826324
12000	16.28	1348.2245	9.362670028	484.1895442
12500	14.5	1321.9343	9.180099552	474.7479303
13000	12.72	1296.0609	9.000422585	465.455954
13500	10.94	1270.599	8.823603914	456.3117941
14000	9.16	1245.5436	8.649608622	447.3136449
14500	7.38	1220.8899	8.478402088	438.4597161
15000	5.6	1196.6328	8.309949987	429.7482326
15500	3.82	1172.7674	8.14421829	421.1774344
16000	2.04	1149.2889	7.981173256	412.7455768
16500	0.26	1126.1925	7.820781438	404.45093
17000	-1.52	1103.4734	7.663009679	396.2917793
17500	-3.3	1081.1268	7.507825109	388.2664247
18000	-5.08	1059.1481	7.355195145	380.3731814

Appendix B cont.

	18500	-6.86	1037.5326	7.205087489	372.610379
	19000	-8.64	1016.2757	7.057470127	364.9763619
	19500	-10.42	995.37283	6.91231133	357.4694892
	20000	-12.2	974.81947	6.769579648	350.0881345
	20500	-13.98	954.61112	6.62924391	342.8306859
	21000	-15.76	934.74334	6.491273226	335.6955458
	21500	-17.54	915.21173	6.355636981	328.6811309
	22000	-19.32	896.0119	6.222304837	321.7858724
	22500	-21.1	877.13953	6.09124673	315.0082155
	23000	-22.88	858.59033	5.962432867	308.3466195
	23500	-24.66	840.36006	5.83583373	301.7995578
	24000	-26.44	822.44449	5.711420069	295.3655177
	24500	-28.22	804.83946	5.589162902	289.0430006
	25000	-30	787.54083	5.469033516	282.8305214
	25500	-31.78	770.5445	5.351003464	276.726609
	26000	-33.56	753.84642	5.235044562	270.729806
	26500	-35.34	737.44256	5.12112889	264.8386684
	27000	-37.12	721.32895	5.009228792	259.051766
	27500	-38.9	705.50163	4.899316869	253.3676819
	28000	-40.68	689.9567	4.791365982	247.7850126
	28500	-42.46	674.69029	4.685349253	242.3023681
	29000	-44.24	659.69857	4.581240056	236.9183713
	29500	-46.02	644.97773	4.479012022	231.6316588
	30000	-47.8	630.52402	4.378639035	226.4408798
	30500	-49.58	616.33371	4.280095233	221.344697
	31000	-51.36	602.40312	4.183355004	216.3417857
	31500	-53.14	588.72859	4.088392983	211.4308343
	32000	-54.92	575.3065	3.995184058	206.610544
	32500	-56.7	562.13328	3.90370336	201.8796289
	33000	-58.48	549.20538	3.813926266	197.2368155
	33500	-60.26	536.51929	3.7258284	192.6808431
	34000	-62.04	524.07153	3.639385625	188.2104636
	34500	-63.82	511.85866	3.554574047	183.8244414
	35000	-65.6	499.87728	3.471370013	179.5215531
	35500	-67.38	488.12402	3.389750108	175.3005879
Aeroboom Deployed	**36000**	**-69.16**	**476.59553**	**3.309691154**	**171.160347**
	36500	-70	462.80546	3.213926772	166.2079016
	37000	-70	451.83035	3.13771078	162.2663992

Appendix B cont.

	37500	-70	441.11552	3.063302196	158.4183667
	38000	-70	430.65477	2.990658158	154.6615876
	38500	-70	420.4421	2.919736823	150.9938978
	39000	-70	410.47162	2.850497336	147.4131847
	39500	-70	400.73757	2.782899815	143.9173857
	40000	-70	391.23437	2.716905321	140.504487
	40500	-70	381.95652	2.65247584	137.1725228
	41000	-70	372.89869	2.589574258	133.9195738
	41500	-70	364.05567	2.528164342	130.7437662
	42000	-70	355.42234	2.468210719	127.6432705
	42500	-70	346.99375	2.409678854	124.616301
	43000	-70	338.76504	2.35253503	121.6611138
	43500	-70	330.73147	2.296746332	118.7760069
	44000	-70	322.88841	2.242280623	115.9593182
	44500	-70	315.23134	2.189106529	113.2094252
	45000	-70	307.75585	2.137193422	110.5247441
	45500	-70	300.45764	2.086511397	107.9037282
	46000	-70	293.3325	2.03703126	105.3448679
	46500	-70	286.37633	1.988724509	102.8466891
Minimum Altitude	**47000**	**-70**	**279.58512**	**1.941563319**	**100.4077529**
	47500	-70	272.95496	1.895520523	98.02665429
	48000	-70	266.48202	1.850569599	95.70202176
	48500	-70	260.16259	1.806684654	93.43251624
	49000	-70	253.99302	1.76384041	91.21683043
	49500	-70	247.96975	1.722012187	89.05368804
	50000	-70	242.08933	1.68117589	86.94184303
	50500	-70	236.34835	1.641307997	84.88007893
	51000	-70	230.74352	1.602385543	82.8672081
	51500	-70	225.2716	1.564386107	80.90207108
	52000	-70	219.92944	1.527287801	78.98353588
	52500	-70	214.71397	1.491069255	77.11049739
	53000	-70	209.62218	1.455709605	75.28187667
	53500	-70	204.65114	1.421188485	73.4966204
	54000	-70	199.79799	1.387486009	71.75370021
	54500	-70	195.05992	1.354582763	70.05211214
	55000	-70	190.43421	1.322459794	68.39087602
	55500	-70	185.9182	1.291098599	66.76903494
	56000	-70	181.50928	1.260481112	65.18565466

Appendix B cont.

	56500	-70	177.20492	1.230589697	63.63982313
	57000	-70	173.00263	1.201407136	62.13064989
	57500	-70	168.89999	1.172916618	60.65726563
	58000	-70	164.89465	1.145101733	59.21882162
	58500	-70	160.98429	1.117946458	57.8144893
	59000	-70	157.16666	1.091435152	56.44345972
	59500	-70	153.43957	1.065552542	55.10494314
	60000	-70	149.80086	1.040283719	53.79816852
	60500	-70	146.24843	1.015614129	52.52238314
	61000	-70	142.78026	0.991529561	51.2768521
	61500	-70	139.39432	0.968016141	50.06085795
	62000	-70	136.08869	0.945060326	48.87370023
	62500	-70	132.86144	0.922648891	47.71469512
	63000	-70	129.71073	0.900768927	46.58317499
	63500	-70	126.63473	0.879407831	45.47848806
	64000	-70	123.63168	0.858553299	44.39999799
	64500	-70	120.69984	0.838193317	43.34708355
	65000	-70	117.83753	0.818316157	42.31913823
	65500	-70	115.04309	0.79891037	41.31556991
	66000	-70	112.31493	0.779964778	40.33580049
	66500	-70	109.65146	0.761468467	39.37926561
	67000	-70	107.05115	0.743410782	38.44541428
	67500	-70	104.51251	0.725781323	37.53370856
	68000	-70	102.03407	0.708569934	36.6436233
	68500	-70	99.614405	0.691766701	35.77464579
	69000	-70	97.25212	0.675361945	34.92627545
	69500	-70	94.945855	0.659346216	34.09802362
	70000	-70	92.694282	0.643710288	33.28941319
	70500	-70	90.496102	0.628445156	32.49997838
	71000	-70	88.350052	0.613542025	31.72926446
	71500	-70	86.254893	0.598992311	30.97682748
	72000	-70	84.209419	0.584787634	30.242234
	72500	-70	82.212453	0.57091981	29.52506088
	73000	-70	80.262843	0.557380852	28.82489502
	73500	-70	78.359466	0.544162961	28.14133309
	74000	-70	76.501227	0.531258522	27.47398135
	74500	-70	74.687055	0.518660103	26.82245539
Scheduled Aeroboom Deployment	**75000**	**-70**	**72.915904**	**0.506360447**	**26.18637989**

Appendix B cont.

75500	-70	71.186755	0.494352469	25.56538848
76000	-70	69.498612	0.482629251	24.95912344
76500	-70	67.850502	0.471184041	24.36723554
77000	-70	66.241475	0.460010245	23.78938384
77500	-70	64.670606	0.449101429	23.22523549
78000	-70	63.136988	0.438451307	22.67446552
78500	-70	61.639739	0.428053746	22.13675666
79000	-70	60.177997	0.417902755	21.61179919
79500	-70	58.750918	0.407992488	21.09929071
80000	-70	57.357682	0.398317236	20.598936
80500	-70	55.997485	0.388871425	20.11044686
81000	-70	54.669545	0.379649615	19.63354188
81500	-70	53.373095	0.370646494	19.16794637
82000	-70	52.10739	0.361856875	18.71339213
82500	-69.75	52.01555	0.361219096	18.6804094
83000	-68.93	50.786003	0.352680575	18.23884066
83500	-68.11	49.588006	0.344361156	17.80860275
84000	-67.29	48.420688	0.336254776	17.38938209
84500	-66.47	47.283199	0.32835555	16.98087443
85000	-65.65	46.174719	0.32065777	16.58278451
85500	-64.83	45.094449	0.313155895	16.1948258
86000	-64.01	44.041615	0.305844548	15.81672023
86500	-63.19	43.015466	0.29871851	15.4481979
87000	**-62.37**	**42.015271**	**0.291772716**	**15.08899685**
87500	-61.55	41.040324	0.285002249	14.7388628
88000	-60.73	40.089936	0.278402335	14.39754892
88500	-59.91	39.163441	0.271968341	14.06481556
89000	-59.09	38.260191	0.265695768	13.74043007
89500	-58.27	37.379556	0.259580248	13.42416657
90000	-57.45	36.520926	0.25361754	13.11580573
90500	-56.63	35.683708	0.247803526	12.81513457
91000	-55.81	34.867326	0.242134206	12.52194626
91500	-54.99	34.07122	0.236605697	12.23603996
92000	-54.17	33.294849	0.231214226	11.95722058
92500	-53.35	32.537683	0.22595613	11.68529866
93000	-52.53	31.79921	0.22082785	11.42009017
93500	-51.71	31.078934	0.215825929	11.16141632
94000	-50.89	30.376369	0.210947009	10.90910348

Average Burst Altitude

Appendix B cont.

94500	-50.07	29.691047	0.206187828	10.6629829
95000	-49.25	29.022511	0.201545216	10.4228907
95500	-48.43	28.370318	0.197016094	10.18866759
96000	-47.61	27.734036	0.192597469	9.960158849
96500	-46.79	27.113246	0.188286434	9.73721409
97000	-45.97	26.507543	0.184080162	9.519687192
97500	-45.15	25.916531	0.179975909	9.307436143
98000	-44.33	25.339825	0.175971005	9.100322925
98500	-43.51	24.777051	0.172062856	8.898213394
99000	-42.69	24.227848	0.168248941	8.700977161
99500	-41.87	23.691861	0.164526809	8.508487481
100000	-41.05	23.168747	0.160894078	8.320621147
100500	-40.23	22.658174	0.157348431	8.13725838
101000	-39.41	22.159817	0.153887617	7.958282731
101500	-38.59	21.67336	0.150509447	7.78358098
102000	-37.77	21.198498	0.147211791	7.613043041
102500	-36.95	20.734932	0.14399258	7.44656187
103000	-36.13	20.282371	0.140849801	7.284033375
103500	-35.31	19.840536	0.137781497	7.125356327
104000	-34.49	19.40915	0.134785763	6.970432281
104500	-33.67	18.987948	0.13186075	6.819165488
105000	-32.85	18.57667	0.129004655	6.671462821
105500	-32.03	18.175065	0.126215727	6.527233696
106000	-31.21	17.782886	0.123492262	6.386389999
106500	-30.39	17.399895	0.120832604	6.248846014
107000	-29.57	17.02586	0.118235138	6.114518351
107500	-28.75	16.660555	0.115698297	5.983325882
108000	-27.93	16.30376	0.113220555	5.855189675
108500	-27.11	15.955261	0.110800426	5.730032928
109000	-26.29	15.614851	0.108436464	5.60778091
109500	-25.47	15.282326	0.106127265	5.488360899
110000	-24.65	14.95749	0.103871459	5.37170213

Appendix C: List of HAPCAD Components

HAPCAD Component	**Purpose**	**Technical Name**	**Flight**
ELECTRICAL COMPONENTS			
Nucleo	Main computer	mbed Nucleo-F401RE	1 &2
Lithium Polymer Battery	Power source	E-flite 3S 11.1V 1800mAh	1 & 2
SD Card Data Saver	Data Management	SparkFun microSD_Shield_v14	1
Temperature Sensors	Environment Sensor	SparkFun TMP36	1 & 2
OpenLog DataLogger	Data Management		2
GPS	Location Tracking	SparkFun EM-506 GPS Shield-v16	2
GoPro Camera	Video Recording	GoPro HERO Camera	1 & 2
Camera **x2**	Image Capture		1 & 2
HARWARE COMPONENTS			
Dyneema Fishing Line	Aeroboom Tie Up	Dyneema Fiber	1 & 2
HARWARE COMPONENTS BUILT IN-HOUSE			
AeroBoom	Main Payload	GAS AeroBoom Team	1 & 2
Scientific Board	Manage Sensors	Jorden Luke	1 & 2
Nichrome Wire Cutters	Deployment Mechanism	James Gardiner	1 & 2
Camera Frame	Hold Payload and Cameras	HARBOR Team	1 & 2

Links to Component Websites:

Nucleo: https://developer.mbed.org/platforms/ST-Nucleo-F401RE/

Lithium Polymer Battery (about): http://www.rogershobbycenter.com/lipoguide/

SD Shield: https://www.sparkfun.com/products/12761

Temperature Sensors: https://www.sparkfun.com/products/10988

OpenLog DataLogger: https://www.sparkfun.com/products/9530

GPS: https://www.sparkfun.com/products/13199

GoPro: http://shop.gopro.com/cameras/hero/CHDHA-301.html

Dyemma: http://www.dsm.com/products/dyneema/en_US/product-technologies/fiber.html

Appendix D: Nucleo Code

Flight #1 Nucleo Code:

```
//**********Included Libraries**********
#include "mbed.h"
#include "Adafruit_ADS1015.h"
#define SERIAL_BAUD_RATE    9600

//************Pin Assinments*************
I2C i2c(I2C_SDA, I2C_SCL);
Adafruit_ADS1115 ads(&i2c);         // ADS115 AD Converter
Serial pc(SERIAL_TX, SERIAL_RX);    // USB CDC serial port
DigitalOut myled(LED1);             // Board LED for system progress indication
DigitalOut NCWC1(PH_0);             // Nichrome wire cutter #1
DigitalOut NCWC2(PH_1);             // Nichrome wire cutter #2
AnalogIn Temp1(A1);                 // External Temp Sensor TMP36
DigitalIn RBF (A0);                 // Remove before flight Pin
AnalogIn UV(A3);                    // External UV Sensor
DigitalIn mybutton(USER_BUTTON);    // User Button
Timer t;                            // Timer
DigitalIn MSA_Sensor (A2);          // HARBOR Flight MSA
DigitalOut Heater(A4);              // AeroBoom Heater
AnalogIn Seperation_Switch (A5);    // Aeroboom Lid Seperation Switch
Serial openlog(PA_11,PA_12);        // Data Logger


//************* Main Program ***************
int main() {

 //********** Initial Settings *************
       NCWC1= 0;
       NCWC2= 0;
       Heater= 0;
       ads.setGain(GAIN_SIXTEEN); // set range to +/-0.256V
 //************** Start Up *****************
   while(RBF.read() == 0){
       myled = !myled;        // Holding loop before RBF is removed and the flight starts
       wait(1);
   }
    // File Open Open Log*****
    printf("RBF REMOVED TIMER START");
    openlog.printf("RBF REMOVED TIMER START");
    t.start();
```

```
//*********** Flight Program *************
     float Seperation_Switch ;
      Seperation_Switch = ((reading1 *3.3);       // separation switch reading/ state
     int Deploy;

 if (  MSA_Sensor==1 && t >= 120 && t <= 200 && Deploy==0){
          NCWC1= 1;                   //
          wait(10);                  // Initial Attempt to Deploy AeroBoom
          NCWC1= 0;                   //
          printf("NCWC1 Activation");
          openlog.printf("NCWC1 Activation");
             if(Seperation_Switch==1){
                // Camera Picture****
                myled=1;
                Deploy=1;                   // AeroBoom sucessfull deployment
                printf("Boom Deployed 1");
             }else{
                NCWC2= 1;                   //
                wait(30);                  //  Secondary attempt to deploy AeroBoom ( NCWC2 )
                NCWC2= 0;                   //  ( note longer heating time )
                printf("NCWC2 Activation");
                openlog.printf("NCWC2 Activation");
                   NCWC1= 1;               //
                   wait(30);               //  Secondary attempt to deploy AeroBoom with NCWC1
                   NCWC1= 0;               //  ( note longer heating time )
                   printf("NCWC1 Activation2");
                   openlog.printf("NCWC1 Activation2");
                // Camera Picture*****
                myled=1;
                Deploy=1;
                printf("Boom Deployed 2");
                openlog.printf("Boom Deployed 2");
                   if  (Seperation_Switch==0){
                      printf("Deployment Failed No Lid Seperation Detected");
                      openlog.printf("Deployment Failed No Lid Seperation Detected");

                      }
                }
}else{
      //sensor data//
      int reading1 = ads.readADC_SingleEnded(0); // read channel 0 on I2c AD converter
      int reading2 = ads.readADC_SingleEnded(1); // read channel 1 on I2c AD converter
      int reading3 = ads.readADC_SingleEnded(2); // read channel 2 on I2c AD converter
      int reading4 = ads.readADC_SingleEnded(3); // read channel 3 on I2c AD converter
```

```
    float Pressure_sensor ;
     //conversion to degrees C - from sensor output voltage per TMP36 data sheet
     Pressure_sensor = ((reading *3.3)-0.500)*100.0;
    float AeroBoom_temp ;
     //conversion to degrees C - from sensor output voltage per TMP36 data sheet
     AeroBoom_temp = ((reading2 *3.3)-0.500)*100.0;
    float Camera_temp ;
     //conversion to degrees C - from sensor output voltage per TMP36 data sheet
     Camera_temp = ((reading3 *3.3)-0.500)*100.0;
                                    //******Data Recording Text*****
   pc.printf("Time:%d,Pressure(torr):%d,AeroBoom Temp:%d,Camera Temp:%d,Extern
Temp:%d,UV Sensor:%d,Separation Switch:%d
\r\n",t,Pressure_sensor,AeroBoom_temp,Camera_temp,Temp1,UV,Seperation_Switch);
openlog.printf("Time:%d,Pressure(torr):%d,AeroBoom Temp:%d,Camera Temp:%d,Extern
Temp:%d,UV Sensor:%d,Separation Switch:%d
\r\n",t,Pressure_sensor,AeroBoom_temp,Camera_temp,Temp1,UV,Seperation_Switch);



     if ( AeroBoom_temp <= 10) {                    //
        Heater=1;                        //
           printf(" AeroBoom Heater on");           //
           openlog.printf("AeroBoom Heater on");      //
     }else{                              //   AeroBoom Heater Loop
           Heater=0;                     //
           printf(" AeroBoom Heater off");          //
           openlog.printf("AeroBoom Heater off");     //
           }                             //
           wait(30);                          // Timed Data Recording Delay (sec)
   }
}
```

Flight #2 Nucleo Code:

```
#include "mbed.h"
#include "SerialGPS.h"




DigitalOut myled(LED1);
DigitalOut NCWC1(A1);
DigitalOut NCWC2(A2);
AnalogIn tempBoard(A4);
AnalogIn tempBoom(A3);
DigitalOut myled2(A0);
```

```
DigitalIn RBF(PA_13);
SerialGPS gps(PA_11, PA_12, 4800);
AnalogIn tempExtern(A5);
AnalogIn tempBoom2(PC_5);



int main() {

    float A,B,C,D, TEMP_BOOM,TEMP_BOOM2,TEMP_EXT, TEMP_Board;
    int ALT =0;
    int Dply =0;
    int trig_time;
    int trigger_time = 3600;
    int TIME_SFE=0;
    int Start=0;
    set_time(0000000000); // Set RTC time to 16 December 2013 10:05:23 UTC
    time_t seconds = time(NULL);
    seconds = time(NULL);
    trig_time = seconds + trigger_time;

     A= (((tempBoom.read()*3.3)-0.600)*100.00);
     A= 20-A;
     B= (((tempBoard.read()*3.3)-0.600)*100.0);
     B= 20-B;
     C= (((tempExtern.read()*3.3)-0.600)*100.00);
     C= 20-C;
     D= (((tempBoom2.read()*3.3)-0.600)*100.00);
     D= 20-D;


printf("%4.2f %4.2f %4.2f %4.2f\n", A,B,C,D);

     printf(" SFE, ALT_SFE, Deploy, Temp_Boom,Temp_Boom2,Ext_Temp, Internal_Temp,
Sats, Long, Lat, Alt, GPS_Time, RTC\n");


while(1) {

     TEMP_BOOM= (((tempBoom.read()*3.3)-0.600)*100.00)+ A;
     TEMP_Board = (((tempBoard.read()*3.3)-0.600)*100.0)+B;
     TEMP_EXT= (((tempExtern.read()*3.3)-0.600)*100.00)+C;
     TEMP_BOOM2= (((tempBoom2.read()*3.3)-0.600)*100.00)+D;
     gps.sample();
```

```
        seconds = time(NULL);




 if (RBF ==1){                     // Flight Mode RBF Removed


         if  (Start<=1){
            set_time(0000000000); // Set RTC time to 16 December 2013 10:05:23 UTC
            seconds = time(NULL);



          }

             Start=2;



      if(gps.sample()) {
                        printf("%d, %d, %d, %4.2f, %4.2f, %4.2f, %4.2f, %d, %4.2f, %4.2f, %4.2f,
%4.2f,
%s\n",TIME_SFE,ALT,Dply,TEMP_BOOM,TEMP_BOOM2,TEMP_Board,TEMP_EXT,
gps.sats, gps.longitude, gps.latitude, gps.alt, gps.time, ctime(&seconds));

                                    if (gps.alt >= 16764){
                                          ALT++;
                                       }

                                    if (seconds >= trig_time){
                                     TIME_SFE= 1;
                                       }

                if  ((ALT>=10 && Dply==0)||(seconds >= trig_time && Dply == 0)){
//Deployment Sequance

                        myled2 =1;
                       NCWC1=1;
                        wait(6);
                       NCWC1=0;
                        myled2 =0;
                        Dply = 1;
                               int DATASAMPLE=0;
                               while(DATASAMPLE<=20){
```

```
                         gps.sample();
                        seconds = time(NULL);
                              TEMP_BOOM= (((tempBoom.read()*3.3)-0.600)*100.00)+ A;
                              TEMP_Board = (((tempBoard.read()*3.3)-0.600)*100.0)+B;
                              TEMP_EXT= (((tempExtern.read()*3.3)-0.600)*100.00)+C;
                              TEMP_BOOM2= (((tempBoom2.read()*3.3)-0.600)*100.00)+D;


                        printf("%d, %d, %d, %4.2f, %4.2f, %4.2f, %4.2f, %d, %4.2f, %4.2f,
%4.2f, %4.2f,
%s\n",TIME_SFE,ALT,Dply,TEMP_BOOM,TEMP_BOOM2,TEMP_Board,TEMP_EXT,
gps.sats, gps.longitude, gps.latitude, gps.alt, gps.time, ctime(&seconds));
                         DATASAMPLE++;
                          wait(4);
                       }
                            wait(1);
                                 printf("End Mission");
                                 wait(1);
                                 deepsleep();
                                 return 0;

            }else{
                 myled2=1;
                 wait(1);
                 myled2=0;
                 wait(1);
                }
 }else{
     printf("%d, X, %d, %4.2f, %4.2f, %4.2f, %4.2f, No Signal, %s\n",TIME_SFE,
Dply,TEMP_BOOM,TEMP_BOOM2,TEMP_Board,TEMP_EXT, ctime(&seconds));
          wait(2);


            if (seconds >= trig_time){
               TIME_SFE= 1;
                }

      if  ((seconds >= trig_time && Dply == 0)){       //Deployment Sequance

           myled2 =1;
           NCWC1=1;
           wait(6);
           NCWC1=0;
           myled2 =0;
```

```
        Dply = 1;

          int DATASAMPLE=0;
          while(DATASAMPLE<=20){
             gps.sample();
              seconds = time(NULL);
                       TEMP_BOOM= (((tempBoom.read()*3.3)-0.600)*100.00)+ A;
                       TEMP_Board = (((tempBoard.read()*3.3)-0.600)*100.0)+B;
                       TEMP_EXT= (((tempExtern.read()*3.3)-0.600)*100.00)+C;
                       TEMP_BOOM2= (((tempBoom2.read()*3.3)-0.600)*100.00)+D;

             printf("%d, X, %d, %4.2f, %4.2f, %4.2f, %4.2f, No Signal, %s\n",TIME_SFE,
Dply,TEMP_BOOM,TEMP_BOOM2,TEMP_Board,TEMP_EXT, ctime(&seconds));

             DATASAMPLE++;

             wait(4);
             }

                   wait(1);
                   printf("End Mission");
                   wait(1);
                   deepsleep();
                   return 0;


      }else{

        myled2=1;
        wait(1);
        myled2=0;
        wait(1);
         }

    }

     }else{           // Stand by Mode RBF Inserted

         if(gps.sample()) {
            printf(" %d, %4.2f, %4.2f, %4.2f, %4.2f\n", gps.sats, gps.longitude, gps.latitude,
gps.alt, gps.time);


               printf("ready\n");
```

```
                myled2 =1;
                wait(0.1);
                myled2 =0;
                wait(0.01);
                myled2 =1;
                wait(0.1);
                myled2 =0;
                wait(0.1);
            } else {
              printf("No lock\n");
              myled2 =1;

            }

        }


    }

}
```